2D Food Printer with UI

Final Report

Submitted to

The Faculty of Operation Catapult XCIX
Rose-Hulman Institute of Technology

Terre Haute, Indiana

Jake Zhang	Solon High School
Jake Zuckerman	Dunlap High School
Luke Cesario	Downers Grove North High School
Yiren Qu	Lyndon Institute

ABSTRACT

Our group aimed to use 3D materials to print pictures and make an edible ink printer. We decided to use raw food material, such as marshmallow fluff and icing which do not need to be heated before being extruded. We used an Arduino, a modern and portable microcontroller, to control two stepper motors to draw pictures. The two stepper motors perform as two axes to simultaneously move to all coordinates of the drawn pictures. The frame of the printer is made of acrylic, protecting the printing area and the breadboard wiring area. We attached two acrylic panels to the linear bearings to create smoother movements from the motors. We created a separate side box to safely house most of the wiring and the Arduino. Our extruder contains an outer protection case, a top plunger and a servo gear controlling the flow of the material. A small plunger is being used to squeeze the edible printing material out of the bag from extruder. Due to the limited memory of Arduino, we used Windows Presentation Form (WPF) C# user interface to make serial port communication, interpret GCode file to points and send them to the Arduino. The two stepper motors work in unison, but we need to test the feasibility of the extruder with different food materials.

1 Introduction

3D printers are becoming an important part in industrial producing, components printing and even food printing. The food printer, a kind of edible ink printing application, transfers images as a thin, edible layer. In our case, we used marshmallow fluff and icing to print on graham crackers or a cake. Food printing provides people with more possibility to make food using the same material especially in art field. It allows us to print high resolution pictures out of a material that is not normally able to be printed.

Our goal is to develop an application of a 2D plotter using edible food material. We used an Arduino as the main microcontroller controlling two stepper motors to move extruders on the printing panel. The method to draw pictures is to move extruder through all of the points with coordinates of the picture. The final product is a picture that is drawn with the space between coordinates.

2 Design

2.1 2D Frame Design

For the frame, we utilized acrylic panels of a quarter inch thickness. The dimensions on the box is 24” x 15” x 6” with a wall placed 5” inside to prevent the griddle from overheating or interrupting the circuits. There is a hole in that wall to allows circuits to run through it for more convenience when it comes to the wiring stage. The sketch of dimension of the box is shown below.

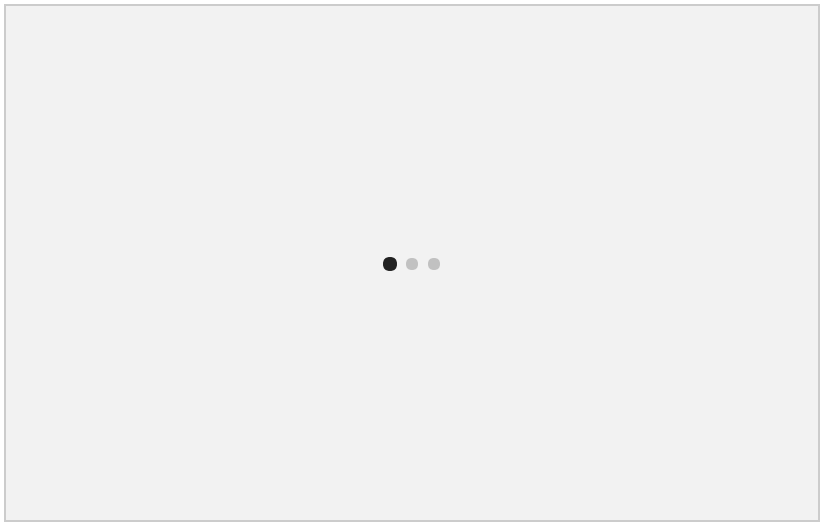

Figure 2.1 Acrylic Frame dimensions sketch

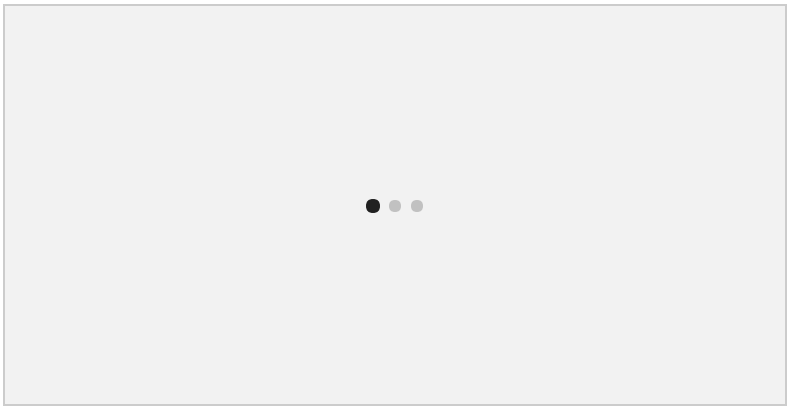

Figure 2.2 Acrylic Middle Panel dimensions sketch

The box was made in this dimension because of the size of the griddle (23” x 11” x 2”) though we are not using the entire surface. Also, the front of the box uses a 2” panel attached to the top, leaving 4” space in the bottom for easy access to the griddle. The front panel has zip tie holes for x-axis stepper motor. The final design is shown below.

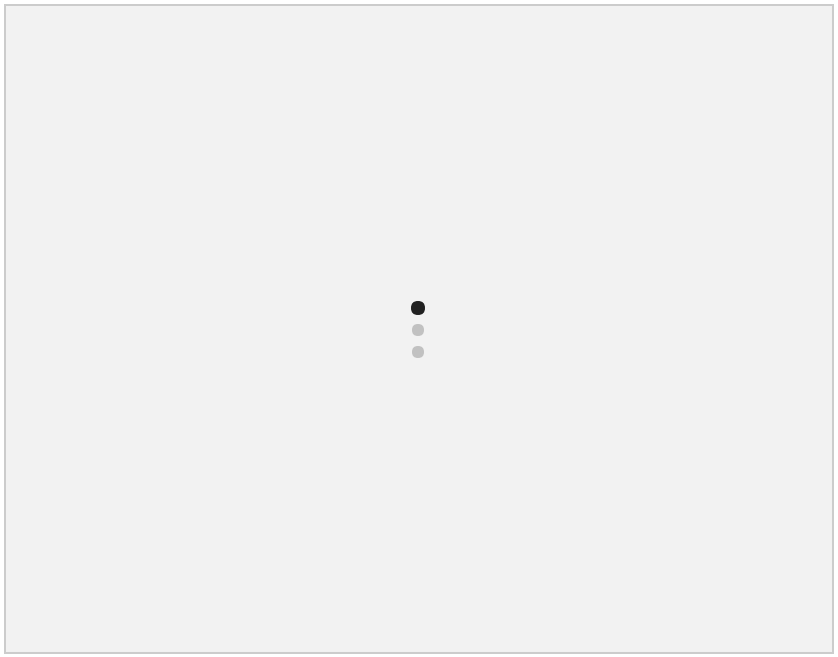

Figure 2.3 Acrylic Final Design sketch

X-axis contains two parallel metal rods and X-axis panel slides on the rods with linear bearings. Also a customized linear bearing holder fix the panel with linear bearings. Y-axis contains another two parallel metal rods and they are fixed on the the X-axis panel. The figure of x-axis and y-axis rods is shown below.

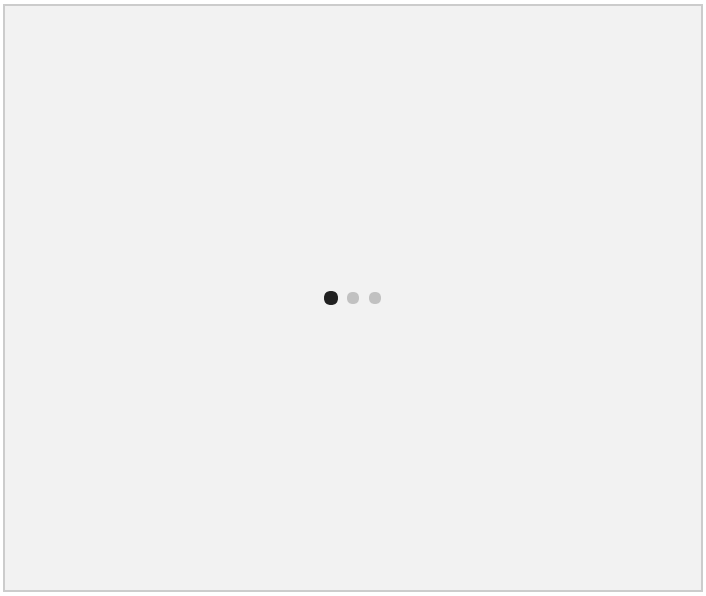

Figure 2.4 Dimensions of Axes rods

The X-axis panel has a middle hole designed for extruder dispenser and center empty space for drawing area. The dimension of x-axis panel is shown below.

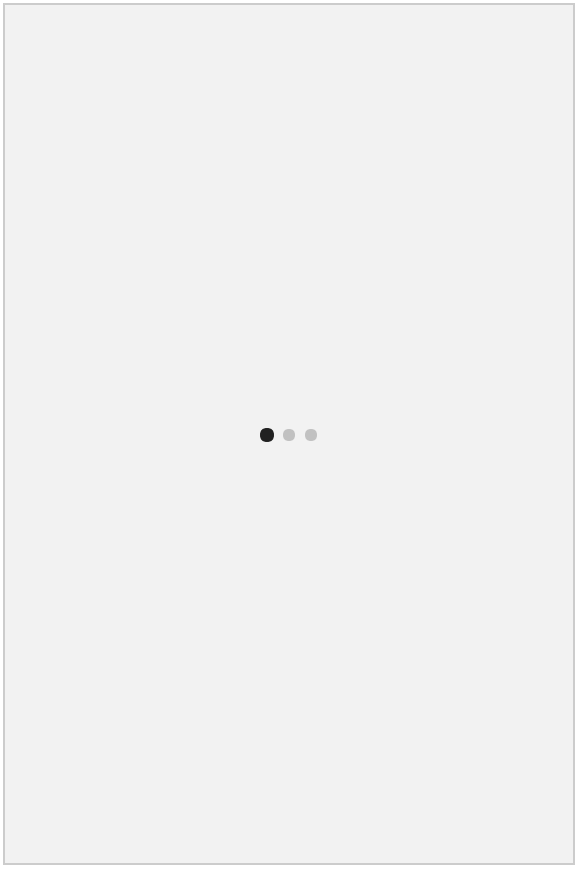

Figure 2.5 Dimensions of X-axis Panel

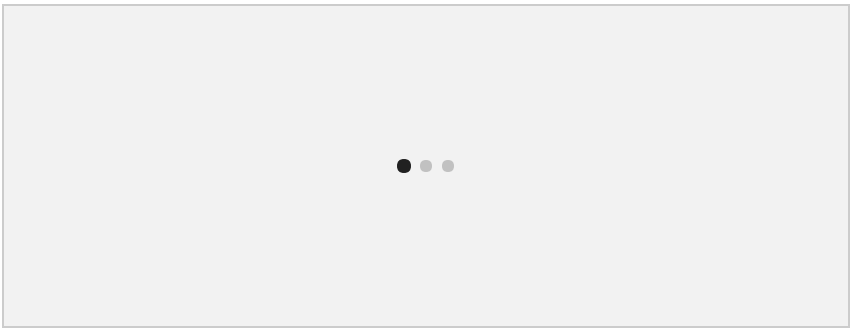

Figure 2.5 Dimensions of Rod Connector

The Y-axis panel connects to the two metal rods on x-axis rods. It has a small circular hole for the extruder dispenser.

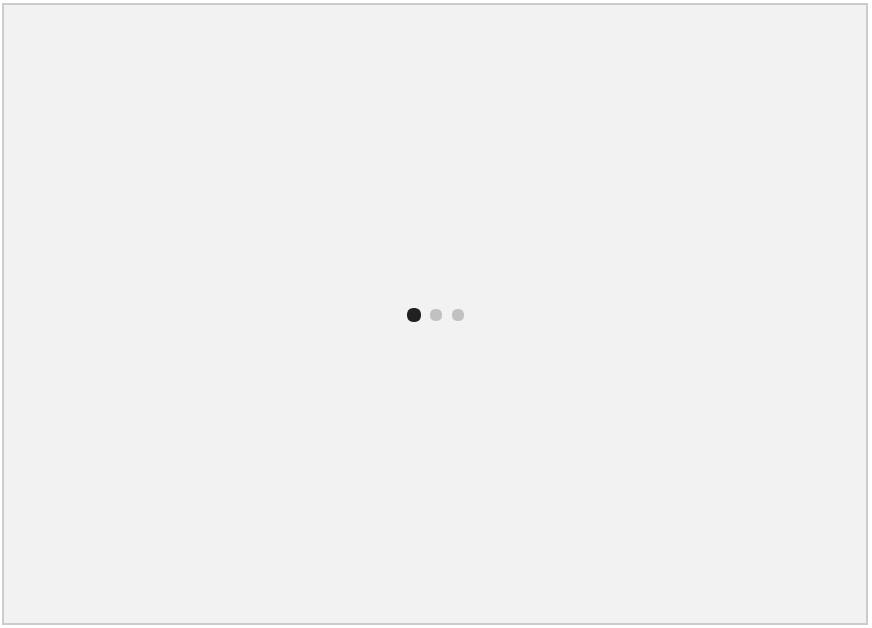

Figure 2.6 Dimensions of Y-axis Panel

2.2 Extruder System

For this system, we used a PVC pipe that is 6 inches tall and approximately 1.5 inches in diameter.

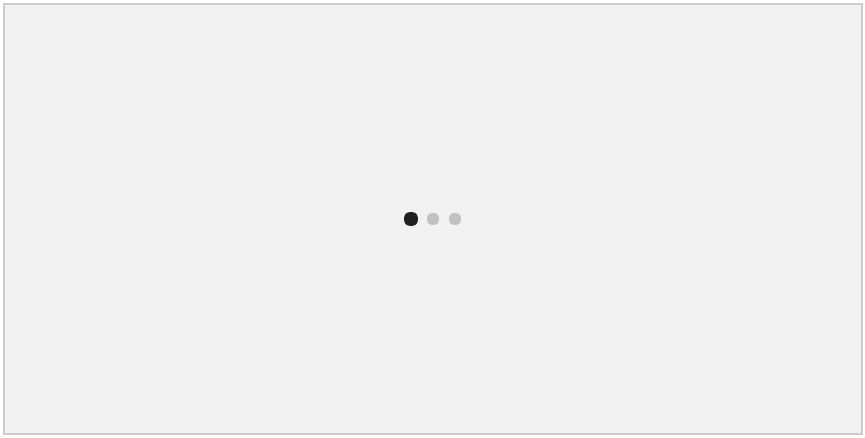

Figure 2.7 Extruder outer dimensions

The Extruder System will hold the bag of icing or marshmallow fluff. But in order to dispense it, we needed a source of pressure to push the materials down out of the bag. We used a circular track ran through a continuous servo and anchor. This pushes the plunger down into the pipe, putting pressure on the bag, and dispensing the material at desirable rate. This pipe was then placed on a 6 x 3 plate with a hole drilled out in the center.

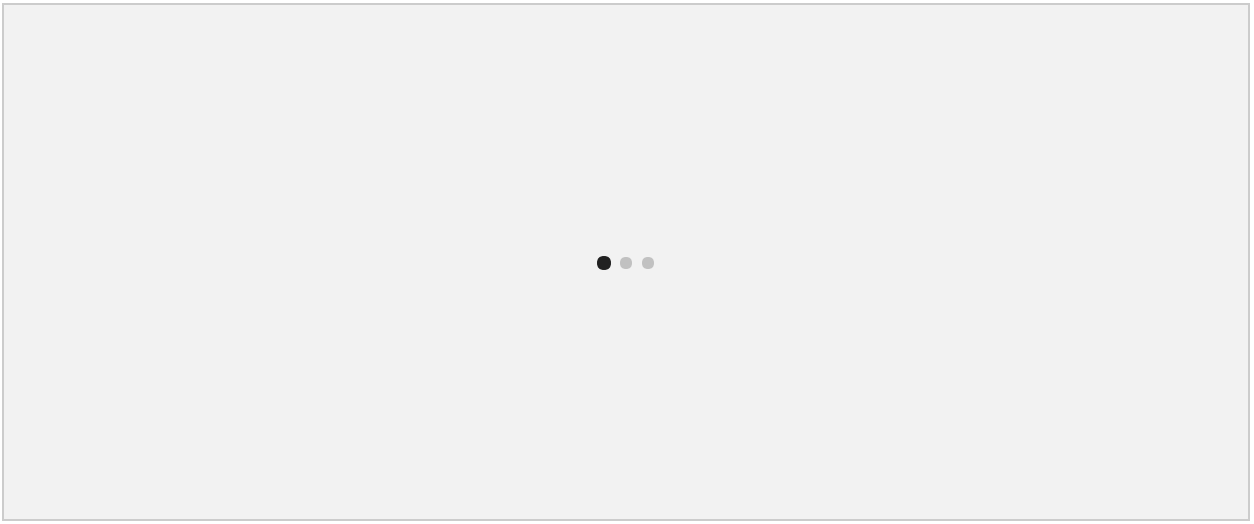

Figure 2.8 Gear system of Extruder

2.3 Serial Communication and UI Design

The Arduino has a limited buffer to store data, consequently there are about 700 points for a normal picture. In order to solve this problem, we decided to use USB communications every point per time to replace sending data altogether. The PC connector, which is built by Windows Presentation Form (WPF) and C#, includes GCode Interpreter and Arduino serial communication

programs. GCode is a type of 2D or 3D printer file which contains X and Y coordinates information and motor control commands. Every time the PC side sends a set of points to the Arduino, it executes the corresponded movement and sends back data to ask the PC to send the next coordinate until the end of the list.

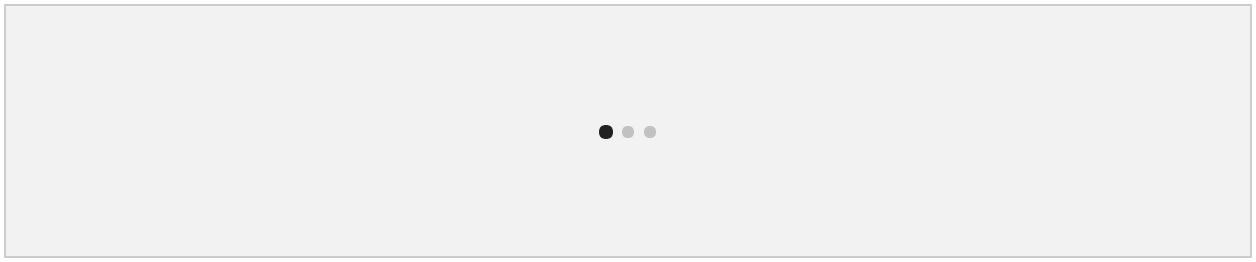

Figure 2.9 Communication System Flow

We used pancake Painter and Inkscape to generate GCode files. The connector we built is to interpret the G-Code and send them to the Arduino.

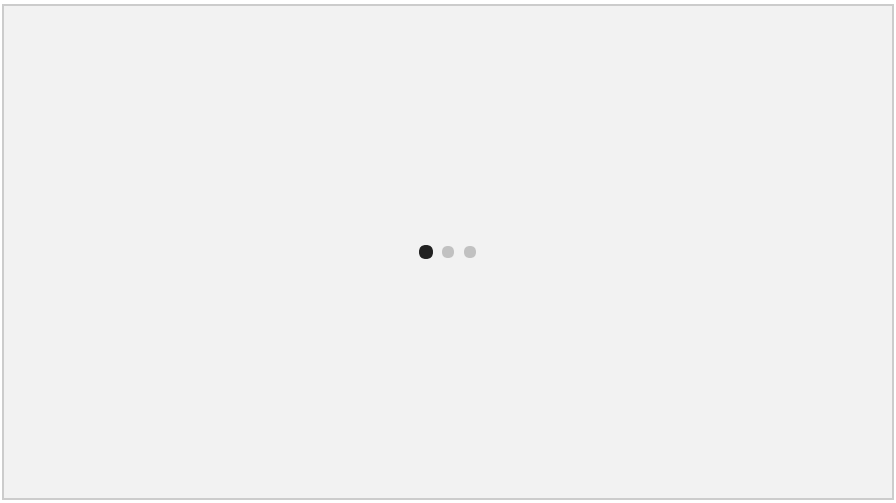

Figure 2.10 Pancake Painter Interface

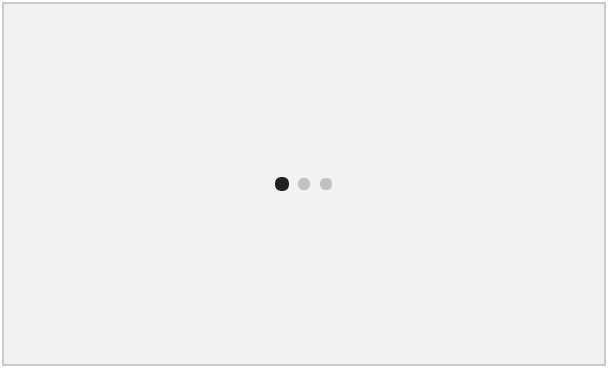

Figure 2.11 InkScape Interface generation GCode for words

The UI Connector has an upload and print function. Also, it plays as a real-time communication system to check the process and time of printing.

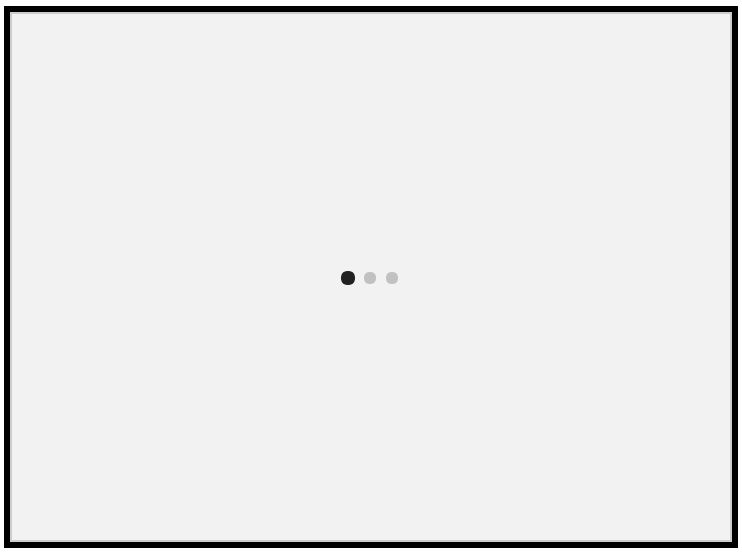

Figure 2.12 UI uploaded GCode

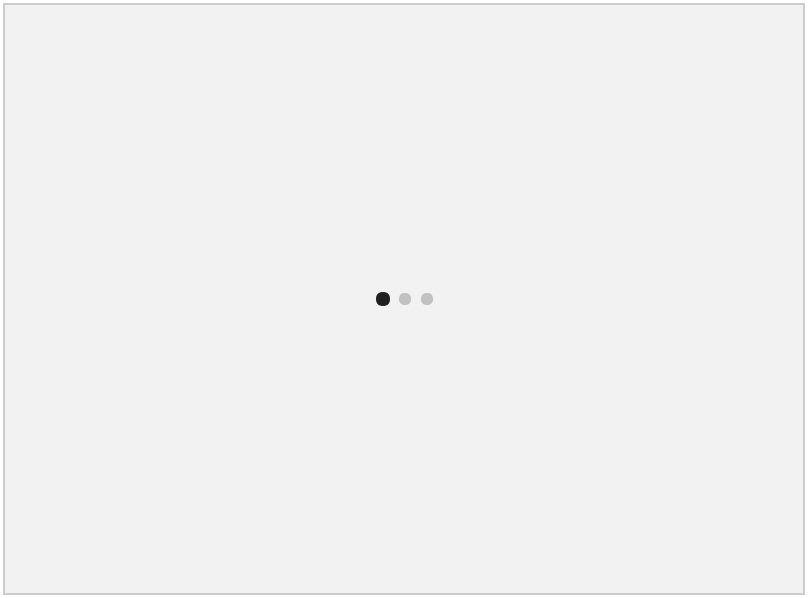

Figure 2.13 Printing GCode to Arduino

2.4 Hardware Design

As seen in Figure 2.14, Our two stepper motors and one servo are being controlled by one Arduino Uno. Also pictured are two Easy-Drivers[1] that allow us to microstep our motors for added precision. The maximum supported current of the step driver is 0.75 amps and the maximum voltage is 20 volts. To stay within these constraints and ensure adequate power, we connected the Arduino to a 12 volt charger.

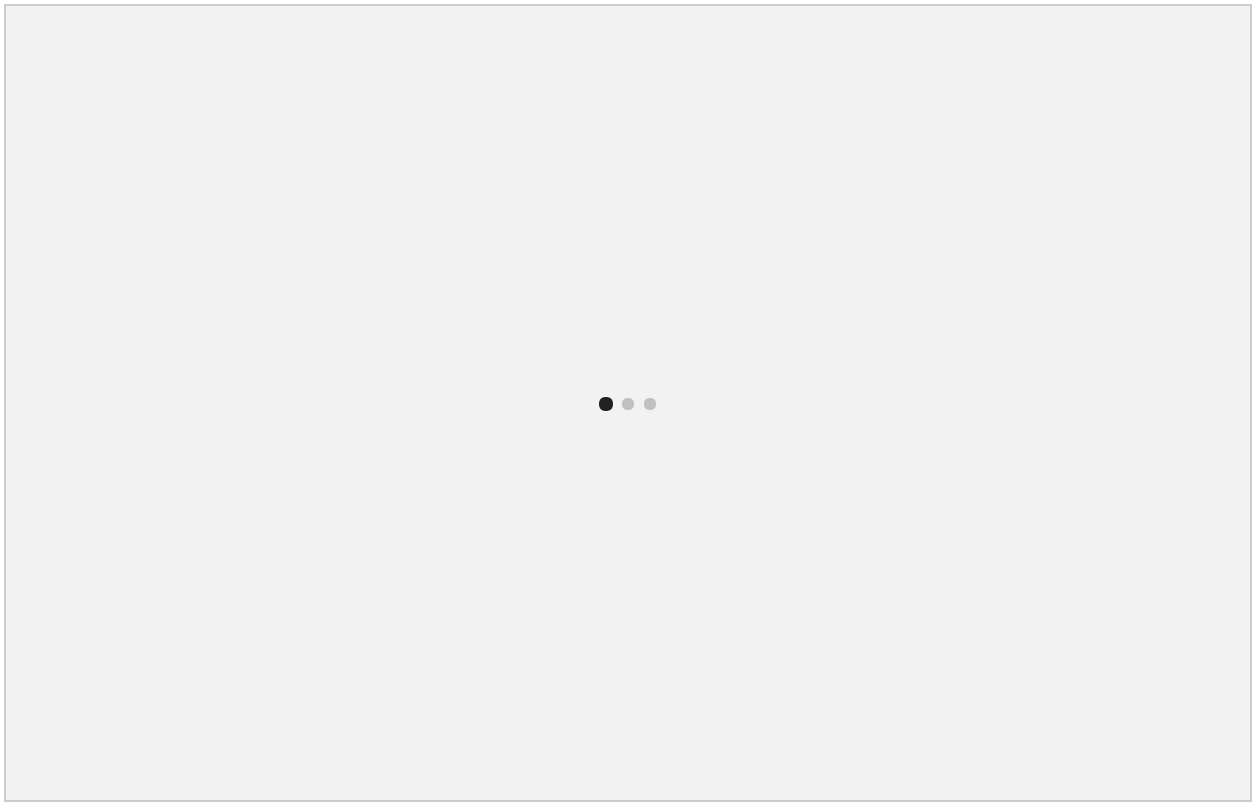

Figure 2.14 Fritzing Diagram

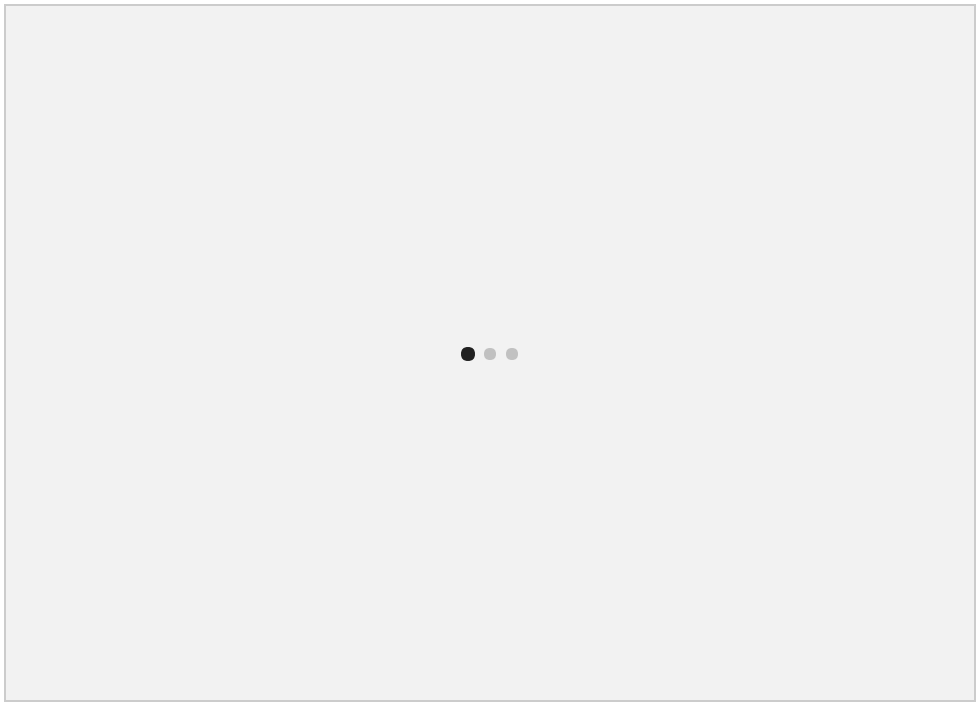

Figure 2.15 Fritzing Schematic

2.5 Arduino Code Design

For the Arduino code, we let it to listen the serial to check the data sent from the PC. the Arduino recognized command and points from the data and execute it. For the stepper motor part, we used Bresenham's line algorithm[2] to rotate the motors by a specific degree amount and we transit distance to certain steps. Because the Arduino does not support threads, we split each motor's required rotations into many smaller rotation steps and alternated between the two motors. Taking this approach has the added benefit of making both motors finish at the same time. For the extruder servo, we used a continuous servo controlled to rotate in different speeds. The Arduino receives commands and then controls the food ink flow.

3 Approach

CONSTRUCTION

For this project, we wanted a large space to work with, especially depending on how much space the griddle took up. Initially we left the frame unglued in case of unavoidable changes. Our team heavily relied on the organization of our roles. We were able to work in an orderly fashion, and it allowed us to cover more ground and finish the project faster. Also, the tight time frame taught us to trust in our teammates.

Our strategy for the construction of the 2D printer involved design in pieces and parts. The plate across the X axis was designed after the axis was attached.

The connectors between panel and linear bearing were printed by 3D printer to make them perfect fit. There were four washer rings on axes rods preventing rods from sliding off during printing.

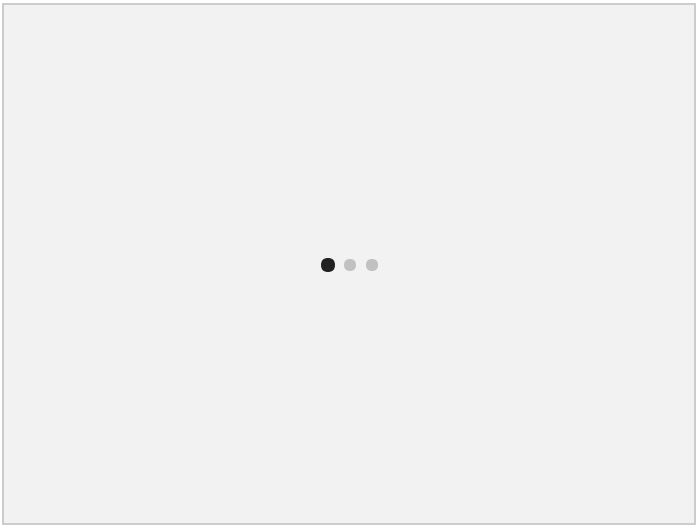

Figure 3.1 Axes and Frame

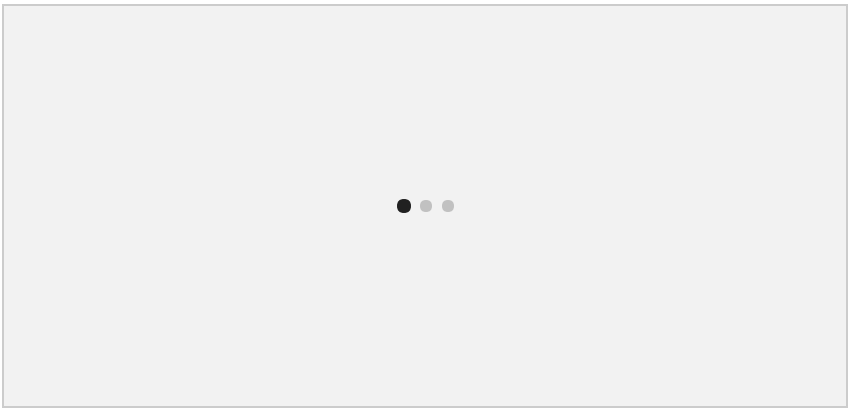

Figure 3.2 Extruder Gear System

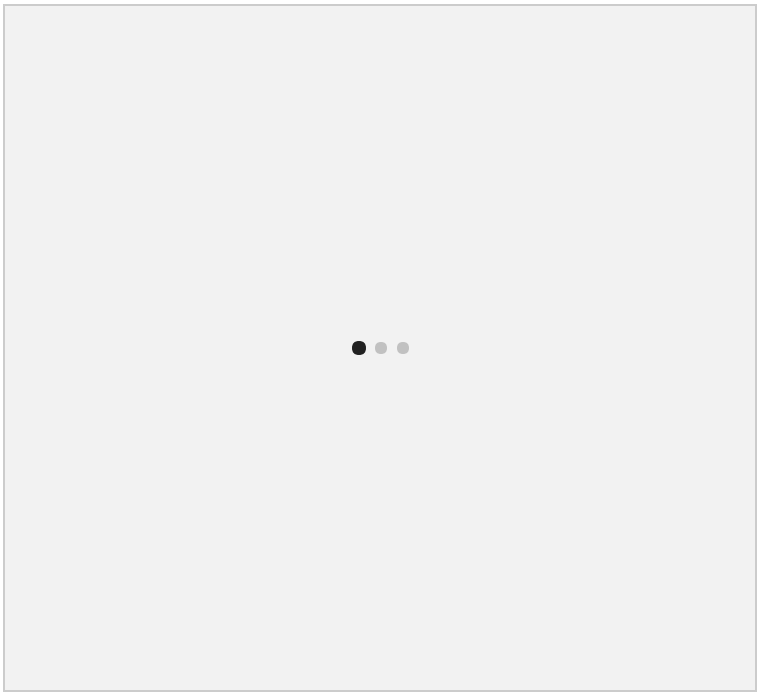

Figure 3.3 Track system

PROGRAMMING

At first, we used a method found on bldr.org to make the stepper motor rotate a certain degree amount. Because we know the dimensions of our axes, and the measurements of the gears on our stepper motors, we can calculate how much each motor needs to rotate to get to the desired location. Our original method for segmenting diagonal lines was very poorly optimized, so we changed it for the better. Now, Bresenham's line algorithm is used to decide which motor needs to step to make a very smooth line. To accommodate this improvement, the required distance is measured in steps. The maximum supported current of the step driver is 0.75 amps and the maximum voltage is 20 volts. To stay within these constraints and ensure adequate power, we connected the Arduino to a 12 volt charger.

To achieve communication between the PC and Arduino, we, according to standard C# communicator of Arduino, built our connector. At first, it sends a message to ask Arduino to start listening to the real command and points. Also, in order to distinguish commands and points, we put ‘C’ before every command and ‘P’ before every point. Every time the PC side sends a set of points to the Arduino, it executes the corresponding movement and sends back data to ask the PC to send the next coordinate until the end of the list.

We used the C# coding language to interpret GCode files and convert them to lists of points in our coordinate system. The command in GCode is shown below in the Table 3.1.

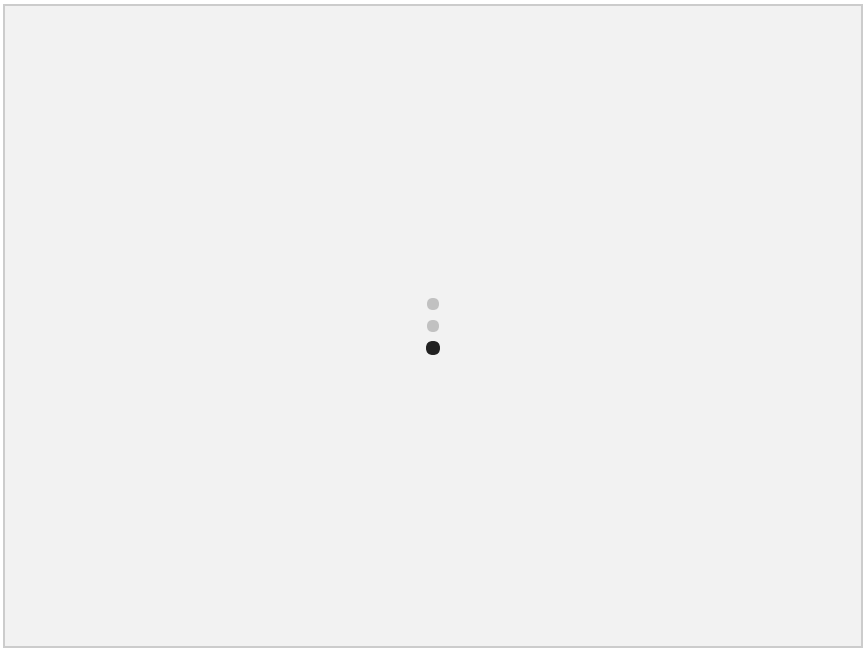

Figure 3.4 Arduino Wiring

COMMAND	MEANING
G04 P(number)	Do nothing for P seconds
G92 [X(number)] [Y(number)]	change logical position
M18	turn off power to motors

Table 3.1 GCode command reference

4 Evaluation

4.1 Communication Test

We used C# serial communication to test the relationship between communication time and the number of points needed to be sent. The data is shown below as Figure 4.1. Their relationship shows a linear direct proportional relationship. Normally a standard picture may have about 700 points. It will not take longer than 8 seconds to communicate between the Arduino and PC.

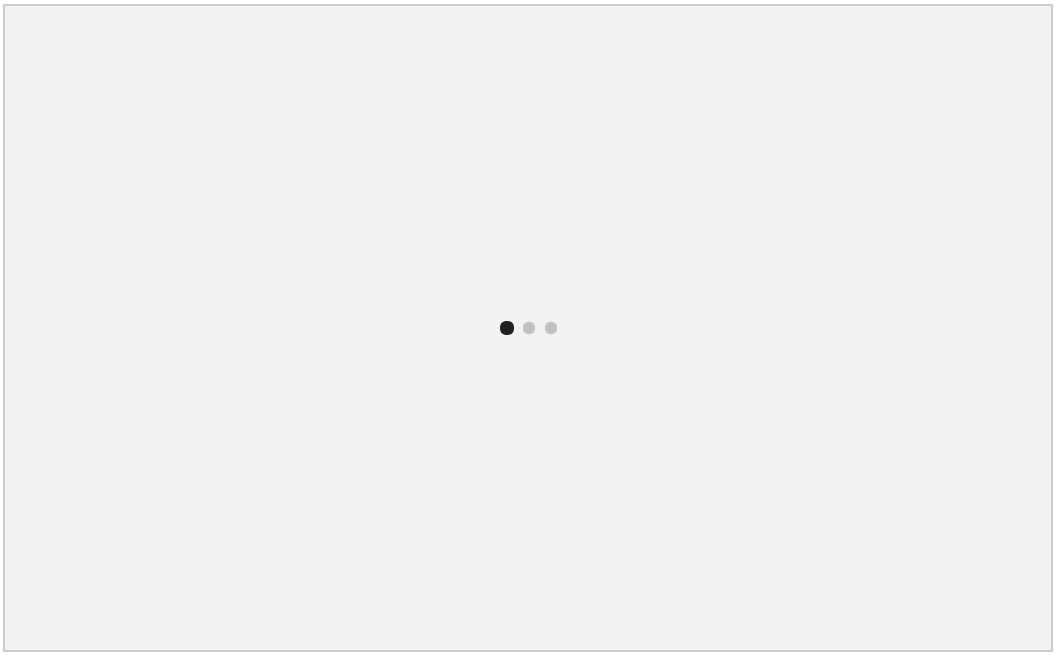

Figure 4.1 Communication time for different points needed to be sent

4.2 Stepper motor Test

Our code draws lines by alternating between the motor for the x axis, and the motor for the y axis. Lines will be different lengths, and our motors can only be so precise so we decided that each line would get segmented differently. This way of handling the segments ensures that the final product will look as it was designed, without looking too choppy.

Achieving this took some troubleshooting. At first, rounding errors caused the rotation necessary to draw a line to be stuck at zero. The very small number of inches per degree of rotation caused this problem. To adapt, we now store the necessary change in rotations that need to be completed, and segment accordingly, minimizing loss of precision. If we need to segment more or less than 360 times, we have to divide the amount of segments and multiply the length of each segment by the same amount to preserve the correct distances.

4.3 Integrated Printer Test

We tested the extruders ability to control flow. It, on average, takes 3 seconds to start squeezing and takes around 6 seconds to stop. Based on the time delay, we set up waiting time every time we started and stopped the squeezing. Because we did not use limit switch on the axis, we

programmed it to move back to (0,0) after passing to all of the points. Also, at the end, the PC program will send a time to let servo release the track and replace the icing bag in the extruder.

5 Conclusion and Future work

5.1 Conclusion

In retrospect, the team learned that, though the construction of the 2D printer seemed like a short task, reality showed us that building the piece can take a lot longer than a couple of days. We were relieved to have programmed the printer before it was done being built, which saved several days time. By the end of the project, our team was content with the results, though there were a few minor changes we would have made had there been more time. We feel this has been a great learning experience for us and has improved our skills in both the computer science and electrical engineering fields.

5.2 Future Plans

As of now, our plans for the 2D printer are mainly based around upgrades. We hope that, in the future, the zip ties used to cable tie the linear bearings will be replaced with something more feasible and long term. We also are hopeful that the application of a 2D food printer will be able to be utilized in the household. On the axes, we can put limit buttons. Every time, when panels push the buttons, the coordinates will reset to (0,0) to prevent coordinate errors. Also, we can keep our original plan by putting a griddle underneath and by using pancake batter to draw pictures on it.

REFERENCES

1 "Use The EasyDriver Stepper Motor Driver Arduino." Weblog post. *Bildr*. Bildr, n.d. Web. 28 June 2016.

2 LaMothe, André. *Tricks of the Windows Game Programming Gurus*. Indianapolis, IN: Sams, 2002. Print.